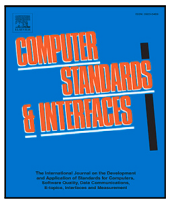
Computer Standards & Interfaces 94 (2025) 104007

Contents lists available at ScienceDirect

Computer Standards & Interfaces

journal homepage: www.elsevier.com/locate/csi



A new deep learning based electricity theft detection framework for smart grids in cloud computing

Zhen Si [a], Zhaoqing Liu [a], Changchun Mu [a], Meng Wang [b], Tongxin Gong [b], Xiaofang Xia [a,*], Qing Hu [c,*], Yang Xiao [d]

[a] School of Computer Science and Technology, Xidian University, Xi'an, 710071, China
[b] School of Computer Science, Xi'an Polytechnic University, Xi'an, 710048, China
[c] School of Environmental Science and Engineering, Southern University of Science and Technology, Shenzhen, 518000, China
[d] Department of Computer Science, The University of Alabama, Tuscaloosa, AL, 35487-0290, USA

ARTICLE INFO

Keywords:
Smart grids
Electricity theft detection
Cloud computing
Deep learning
Auto-correlation mechanism

ABSTRACT

Electricity theft is a widespread problem in smart grids with significant economic and security implications. Although users' electricity consumption patterns usually show obvious periodicity, they also exhibit considerable stochasticity and uncertainty. Existing mainstream electricity theft detection methods are the deep learning-based ones, which struggle to capture reliable long-term dependencies from the complex consumption data, leading to suboptimal identification of abnormal patterns. Moreover, the massive data generated by smart grids demands a scalable and robust computational infrastructure that traditional systems cannot provide. To solve these limitations, we propose a new deep learning-based electricity theft detection framework in cloud computing. At the cloud server, we deploy an electricity theft detector based on the auto-correlation mechanism, called the ETD-SAC detector, which progressively decomposes intricate consumption patterns throughout the detection process and aggregates the dependencies at the subsequence level to effectively discover reliable long-term dependencies from users' electricity consumption data. Experimental results show that the proposed ETD-SAC detector outperforms state-of-the-art detectors in terms of accuracy, false negative rate, and false positive rate.

1. Introduction

In modern smart grids, a combination of advanced technologies — including cloud computing and edge computing — is employed to address varying operational needs [1]. Edge computing is primarily designed for real-time data processing and local pre-processing because of its proximity to data sources and their rapid response capabilities, which make them ideal for immediate decision-making [2]. However, data analysis methods relying on complex deep learning models necessitate extensive storage and significant computational power for reasoning and training. These demands typically surpass the computational and storage capacities of edge devices [3]. Consequently, cloud computing provides a scalable and flexible infrastructure that consolidates substantial storage and computational resources to process, store, and analyze vast amounts of real-time data [4].

Through cloud-based platforms, smart grids can integrate renewable energy sources, enhance demand response programs, and optimize energy usage. This enables utilities to better manage energy distribution efficiently, monitor grid performance, and improve fault detection and resolution, leading to increased grid reliability and reduced operational costs [5]. Substantial evidences indicate that cloud computing already delivers significant cost-effectiveness in smart grid applications. For example, reports in recent years show that enterprises can reduce IT costs by up to 15% by moving to the cloud [6]. This shift not only eliminates the need for expensive servers, but also brings additional savings, such as an average 20% reduction in infrastructure expenses per year. In addition, it also reduces facility and cooling costs. [7].

Electricity theft in smart grids poses a significant challenge globally, with varying levels of prevalence across different countries. Electricity theft incurs an estimated global loss of $96 billion for governments and companies, with around $6 billion lost annually in the United States [8]. Electricity theft is usually much more severe in developing countries. For example, in India, nearly 20% of the total power generated is stolen by malicious users [9]. Many countries have enacted specific laws to combat electricity theft. For example, in South

* Corresponding authors.
E-mail addresses: siz@stu.xidian.edu.cn (Z. Si), zhaoqingliu@stu.xidian.edu.cn (Z. Liu), ccmu@stu.xidian.edu.cn (C. Mu), wangmeng@xpu.edu.cn (M. Wang), 220721091@stu.xpu.edu.cn (T. Gong), xiaxiaofang@xidian.edu.cn (X. Xia), huq@sustech.edu.cn (Q. Hu), yangxiao@ieee.org (Y. Xiao).

https://doi.org/10.1016/j.csi.2025.104007
Received 24 November 2024; Received in revised form 7 March 2025; Accepted 21 March 2025
Available online 31 March 2025

Africa, the Greater Johannesburg Metropolitan Electricity By-laws outline penalties for stealing electricity [10]. However, despite these legal deterrents, electricity theft remains widespread. With methods to tamper with meter readings becoming more versatile, secret, and flexible, electricity theft tends to get even more widespread in modernized power systems [11]. These malicious activities result in economic damage and compromise the stability and reliability of electricity distribution networks, making it imperative to develop robust detection and prevention mechanisms [12].

Existing electricity theft detection (ETD) techniques can be broadly classified into measurement mismatch-based and machine learning-based methods [11]. The fundamental principle of the measurement mismatch-based methods is to continuously narrow down the search area of malicious users until they are ultimately pinpointed [11]. This is primarily achieved by analyzing the readings reported by users' smart meters and the measurements from advanced sensors deployed in the distribution network. However, the high cost of advanced sensors makes it impractical for utility companies to deploy them on a large scale. Therefore, machine learning-based methods have become mainstream detection techniques [13]. The fundamental principle of machine learning-based methods is identifying abnormal electricity consumption patterns highly related to electricity theft, primarily achieved by applying popular machine learning techniques. The most popular detection techniques for electricity theft are those with relatively short detection times and low deployment costs.

As one emerging category of machine learning technologies, transformer-based models use various self-attention mechanisms to flexibly process sequence data and capture complex dependencies, and hence are widely applied in electricity theft detection [14]. For example, the authors in paper [15] introduce an Anomaly Transformer (AT) model, which identifies malicious users by analyzing historical electricity consumption data that deviates from typical consumption patterns. The authors in paper [16] present a Transformer Neural Network (TNN) model, which extracts global features from long-range load sequences alongside local features from segmented parts of the sequence and calculates the relative relationships between these features for user classification. However, although users' electricity consumption patterns usually show obvious periodicity, they also exhibit considerable stochasticity and uncertainty. The intricate temporal patterns of the long-term future prohibit the transformer-based detection techniques from finding reliable dependencies [17], adversely impacting the detection techniques' performance in terms of detection accuracy, false negative rate, and false positive rate.

To address the above limitations, we propose a new deep learning based electricity theft detection framework for smart grids in cloud computing, which comprises a device layer and a cloud service layer. At the device layer, smart meters measure users' electricity consumption data, while the central observer meter measures the total supplied power, and then collects, processes, and transmits the consumption data. At the cloud service layer where the uploaded electricity consumption data are stored and processed, we deploy a new deep learning based electricity theft detector to improve the detection performance of identifying malicious users. The proposed electricity theft detector is based on the series-wise auto-correlation mechanism, called the ETD-SAC detector. It decomposes electricity consumption time series into seasonal and trend components, calculates auto-correlations in the frequency domain using the fast Fourier transform (FFT) for low computational complexity, aggregates dependencies at the subsequence level, and extracts reliable long-range dependencies from users' electricity consumption data. By harnessing the substantial storage capacity and powerful computational resources of cloud servers to store and process electricity consumption data, this proposed framework offers a cost-effective and efficient solution for detecting malicious users conducting electricity theft within smart grids. The main contributions of this paper are summarized as follows:

- We propose a new deep learning based electricity theft detection framework for smart grids in cloud computing, which offers a cost-effective and efficient solution for detecting malicious users.
- We propose the ETD-SAC detector, which is based on the auto-correlation mechanism and capable of effectively discovering reliable long-range dependencies within users' electricity consumption data.
- We conduct extensive experiments to evaluate the ETD-SAC detector. The results show that it surpasses state-of-the-art methods in terms of accuracy, false negative rate, and false positive rate.

We have presented a preliminary short-version of the ETD-SAC detector in a conference paper [18]. The main differences between this paper and the conference version are as follows: Firstly, we provide a systematical and comprehensive literature review in Section 2. Secondly, we demonstrate the new deep learning based electricity theft detection framework in cloud computing in Section 3. Also, we present a clearer introduction of the data flow and implementation details of the proposed ETD-SAC detector in Section 3.3. Finally, we report the simulation experiment and results analysis in Section 4 and draw the conclusion in Section 5.

2. Related works

In this section, we summarize the current methods for electricity theft detection, which, as previously mentioned, can be generally categorized into measurement mismatch-based and machine learning-based methods.

2.1. Measurement mismatch-based methods

As indicated in paper [11], measurement mismatch-based methods aim to identify electricity theft by analyzing discrepancies between reported electricity usage and expected values derived from various sensors and meters within the distribution network. These methods leverage advanced sensors to monitor the electricity flow and detect anomalies indicative of theft. They can be further categorized into sensor deployment-based, group change-based, behavior approximation-based, and control chart-based methods.

The core idea of sensor deployment-based methods is to find an optimal deployment strategy for advanced sensors, such as feeder remote terminal units and digital protective relays. These sensors monitor the electricity flow at strategic points within the distribution network. The goal is to maximize coverage and detection capability while minimizing costs. For instance, a dynamic programming algorithm can determine the minimum number of sensors required to effectively monitor a multi-tenant data center, as seen in paper [19]. The main challenge with this approach is the high cost associated with deploying advanced sensors on a large scale.

The group change-based methods involve installing an inspector box for each community containing a head inspector and several sub-inspectors. The head inspector detects the presence of electricity theft, while the sub-inspectors perform inspection steps on different groups of users. This approach narrows down the search area for malicious users through iterative inspections. A notable example is the group testing-based heuristic inspection algorithm [20], which estimates the ratio of malicious users online and adapts inspection strategies accordingly. The objective is to identify all malicious meters with minimal inspection steps.

As for behavior approximation-based methods, a central observer meter is installed for each community to monitor electricity consumption. Linear or nonlinear functions, or users' behavior functions, are used to model the relationship between actual electricity consumption and reported readings. Tampered meters can be detected by comparing the modeled behavior with actual readings. An example is the use of

Lagrange polynomial interpolation to model an adversary's behavior, allowing for the detection of discrepancies indicative of theft [21].

The control chart-based methods are designed to identify small-scale electricity theft where malicious users manipulate smart meter readings to slightly lower values. By applying cumulative sum control charts and Shewhart control charts together [22], these methods analyze reported readings and measurements from a central observer meter to detect anomalies. However, the method has strict assumptions about the data. Specifically, the data needs to satisfy the normal distribution.

To sum up, measurement mismatch-based methods encounter several significant limitations, which can hinder their practical application. Firstly, they often involve high deployment costs, as implementing such methods can require expensive infrastructure or specialized tools. Secondly, these methods tend to have a relatively long detection time. This implies that they may not be able to identify issues promptly, which is a critical drawback in scenarios where quick detection is essential. Lastly, they are sometimes based on impractical assumptions, which may not hold in real-world environments. This can limit their effectiveness and reduce their applicability in diverse operational conditions.

2.2. Machine learning-based methods

As summarized in paper [11], machine learning-based methods have emerged as a mainstream approach for electricity theft detection due to their ability to model complex and non-linear relationships in the data. These methods focus on identifying abnormal consumption patterns that indicate potential theft. Applying advanced machine learning techniques to analyze meter readings and other customer-related data offers a robust solution for detecting electricity theft.

Among these, deep learning techniques that mimic the human brain's neural networks to enhance decision-making capabilities have led to great breakthroughs in electricity theft detection. For example, the authors in paper [23] explore using deep feedforward, deep recurrent, and deep convolutional-recurrent neural networks to detect electricity theft in distributed generation systems. The authors in paper [24] introduce an ETI-CAE detector, which employs a convolutional autoencoder (CAE) model for electricity theft identification (ETI). The authors in paper [25] present a Deep Neural Network with a Particle Swarm Optimization (LFPR-DNN) detector, which aims to reduce false positive rates in electricity theft detection. However, the above methods cannot effectively capture the periodicity characteristic of users' electricity consumption patterns [13], which adversely impacts the detection accuracy.

To better capture the periodicity characteristics, researchers transform the one-dimensional (1-D) electricity consumption time series by week/month into a two-dimensional (2-D) matrix to emphasize the periodicity of consumption patterns. For example, the authors in paper [26] propose a novel electricity theft detection framework named hybrid-order representation learning network (HORLN), in which the sequential electricity consumption data is transformed into the matrix format containing weekly consumption records. The authors in paper [27] propose a wide and deep convolutional neural network (WDCNN) detector, in which the deep component (a CNN model) is applied to handle matrices containing weekly electricity consumption data. The authors in paper [28] introduce a CNN-LSTM detector in which the CNNs handle monthly electricity consumption data formatted in matrices. However, the above method ignores the inherent temporal dependency in the user's electricity consumption time series, adversely impacting detection accuracy.

Researchers have proposed transformer-based electricity theft detection models to capture temporal dependencies better. For example, the authors in paper [15] develop an anomaly transformer-based model to identify electricity theft. The authors in paper [16] propose a conv-attentional transformer based electricity theft detection approach. However, the above approaches cannot directly identify reliable long-range dependencies from long-term electricity consumption data due to the complexity of consumption patterns.

To address the above limitations, in this paper, we propose a new deep learning-based electricity theft detection framework for smart grids in cloud computing and deploy the proposed ETD-SAC detector based on the autocorrelation mechanism on the cloud server. The ETD-SAC detector progressively decomposes intricate consumption patterns throughout the detection process and aggregates dependencies at the subsequence level to extract reliable long-range dependencies.

3. The proposed ETD framework

In this section, we provide a detailed explanation of the cloud computing-based electricity theft detection framework, in which cloud servers provide huge amounts of computing power and data storage [29], to implement our proposed ETD-SAC detector to detect malicious users within the community. As shown in Fig. 1, the framework primarily consists of the device and cloud service layers. The device layer collects, cleans, and uploads electricity consumption data from all users, along with the total electricity allocated to the community. The cloud service layer stores and processes the uploaded data and then identifies malicious users within the community using the deployed electricity theft detection system. For better understanding, we use a workflow diagram to demonstrate the working strategy of the proposed electricity theft detection framework in Fig. 2. Specifically, the preprocessing step involves filling in missing values and replacing outliers at the device layer. Subsequently, the uploaded electricity consumption data is processed at the cloud server layer to determine the presence of electricity theft and identify malicious users within the community through the proposed ETD-SAC detector.

3.1. The device layer

The device layer consists of smart meters installed for each user and a tamper-proof central observer meter deployed in the community.

As shown in Fig. 1, all the smart meters are connected to the central observer meter via both power flow transmitted through wires and communication flow transmitted through the network. The smart meters periodically measure the corresponding user's electricity consumption and transmit the readings to the tamper-proof central observer meter. The central observer meter receives all smart meters' reported measurements and measures the total power distributed to all users in this community [13,24]. Additionally, the observer meter cleans the collected data by preprocessing technology and uploads them to the cloud server daily. Let $x_{u,t}$ denote user u's reported electricity consumption at period t, with $t \in \{1, 2, 3, \ldots\}$. Let $X_u = \{x_{u,1}, x_{u,2}, \ldots, x_{u,t}, \ldots\}$ denote the time series of user u's reported electricity consumption. The preprocessing process includes filling in missing values and replacing outliers, as demonstrated below.

Filling in missing values: Missing values in users' electricity consumption data may result from smart meter failures, data transmission errors, or storage server malfunctions [30]. To address the problem, we apply a linear interpolation method to fill in the missing values, represented as NaN. Specifically, if a missing value occurs in the second period and its two adjacent values are available, it is replaced with the average of the two adjacent values; otherwise, it is updated to zero. Mathematically, we have

$$x_{u,t} = \begin{cases} x_{u,t}, & x_{u,t} = \text{NaN} \\ \frac{x_{u,t-1}+x_{u,t+1}}{2}, & x_{u,t} = \text{NaN}, x_{u,t-1}, x_{u,t+1} \neq \text{NaN} \\ 0 & \text{otherwise.} \end{cases} \tag{1}$$

Replacing outliers: Transmission errors of smart meters may also lead to outliers in users' electricity consumption data. These outliers indicate significant differences between the electricity consumption of the current period and the adjacent periods. Such outliers will significantly affect the training of deep learning models. Therefore, we follow the work [27] to replace outliers. Specifically, let $avg(X_u)$ denote the mean of user u's time series of electricity consumption data.

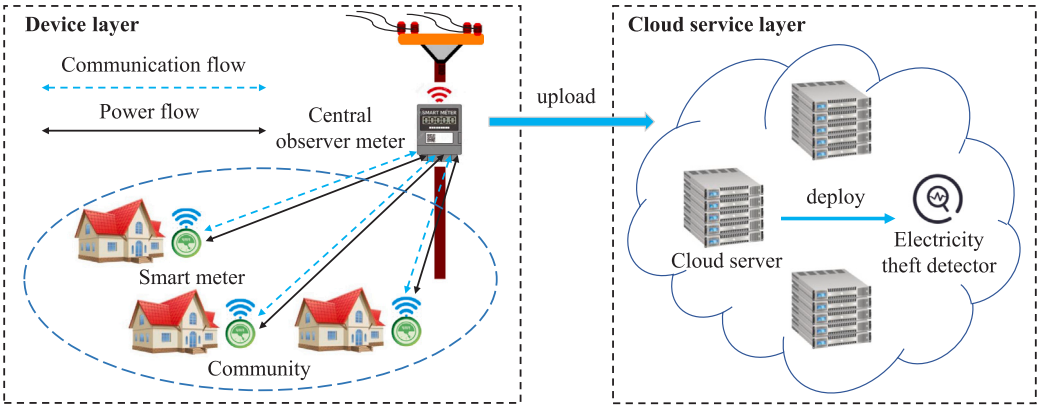


Fig. 1. The deep learning based electricity theft detection framework in cloud computing.

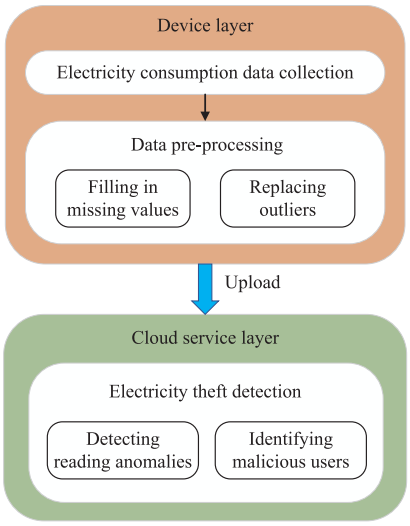


Fig. 2. The working strategy of the proposed electricity theft detection framework in cloud computing.

Let $std(X_u)$ denote the standard deviation of user u's time series of electricity consumption data. Technically, we have

$$x_{u,t} = \begin{cases} avg(X_u)+3std(X_u), & x_{u,t} > avg(X_u)+3std(X_u) \\ x_{u,t}, & \text{otherwise.} \end{cases} \quad (2)$$

Let $R = \{r_1, r_2, \ldots, r_t, \ldots\}$ denote the total electricity consumption measured by the central observer meter, with r_t being the central observer meter's reading at period t. The measurement data is also preprocessed in the same way and then uploaded to the cloud server.

3.2. The cloud service layer

In the cloud service layer, the cloud server mainly performs the following two functions: (1) determining whether there are malicious users in the community by comparing the central observer meter's measurements and the summation of its received readings; (2) if it detects the reading anomalies, i.e., the existence of malicious users, then it applies the trained ETD-SAC detectors to analyze users' electricity consumption readings to identify malicious users, as demonstrated in the following:

(1) Detecting reading anomalies: As well known, technical losses are unavoidable during power transmission and distribution. Let δ_u denote the technical losses of the power transmitted from the central observer meter to user u. In practice, due to the complexity of the smart grid system, it is difficult to establish accurate mathematical models [31]; therefore, existing mathematical models are usually used to estimate the value of δ_u [32]. Based on the law of conservation of energy, if all the users in the community honestly report their electricity consumptions, we have $r_t - \sum x_{u,t} \cong \sum \delta_u$. However, if there are malicious users launching cyber/physical attacks to tamper with the smart meter measurements to smaller values, i.e., to achieve electricity theft purposes, then we have $r_t - \sum x_{u,t} \gg \sum \delta_u$. In this case, the device layer detects reading anomalies, i.e., the existence of malicious users.

(2) Identifying malicious users: Since users' electricity consumption behaviors exhibit weekly periodicity [27], in this paper, the ETD-SAC detectors are trained with input data containing one week of users' electricity consumption data, as shown in Fig. 3. The detector first decomposes the input data into seasonal and trend-cyclical components, after which the seasonal part is transformed into a high-dimensional feature representation. Temporal dependencies are extracted from the seasonal component using the auto-correlation mechanism combined with the trend-cyclical component. By analyzing these temporal dependencies and detecting abnormal consumption patterns of malicious users, the detector generates a predicted label of 0/1 (indicating benign/malicious status) for each day in the input data. The specifics of the ETD-SAC detector deployed on the cloud server are detailed in Sub-Section 3.3.

Additionally, the proposed ETD framework is also designed with scalability in mind to handle data from millions of users in real-world applications efficiently. Key strategies include but are not limited to distributed cloud architecture and batch processing techniques. The distributed cloud architecture implies that the proposed ETD framework employs a distributed cloud computing model, where data processing tasks are distributed across multiple cloud nodes to reduce computational bottlenecks and improve throughput. The batch processing techniques imply that electricity consumption data is processed in batches using parallel computing techniques to enhance efficiency.

Remark: Note that in practical applications, several measures can be implemented to mitigate data privacy concerns when transmitting sensitive user data to cloud servers. Before data transmission, identifiable information like user IDs can be anonymized or replaced with pseudonyms to prevent data linkage to specific individuals. Security protocols like Transport Layer Security (TLS) can be employed to secure data during transmission and mitigate risks such as man-in-the-middle attacks or data interception. For data storage, strong encryption algorithms like AES-256 can be utilized to guard against unauthorized access or insider threats. Simultaneously, strict access permission management can be enforced to ensure that only users with the necessary permissions can access sensitive data.

The details of the deployed ETD-SAC detector on the cloud server are stated as follows.

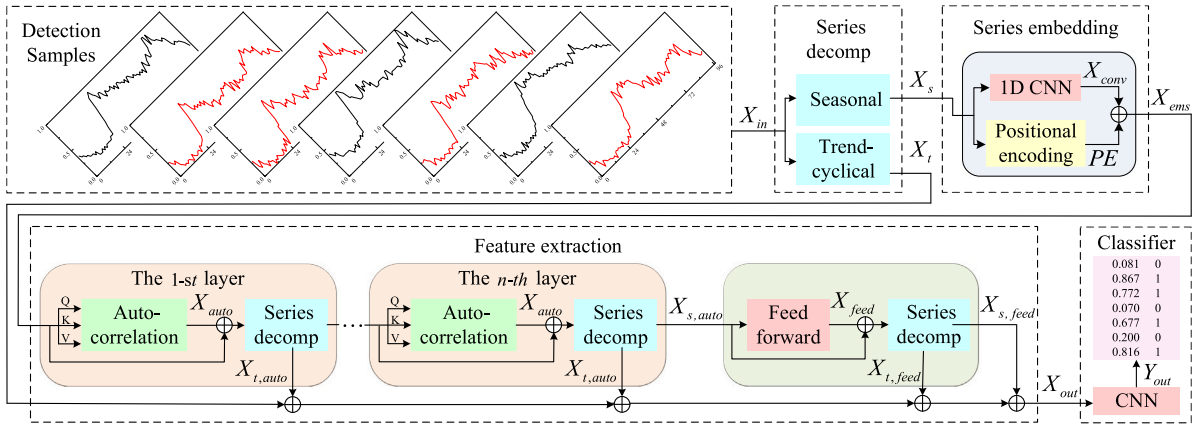


Fig. 3. The overall framework of the proposed ETD-SAC detector.

3.3. The ETD-SAC detector

In this sub-section, we demonstrate the working strategy of the ETD-SAC detector, which mainly consists of four modules: series decomposition, series embedding, feature extraction, and classifier. In the following, if not otherwise specified, we drop the subscript u of $x_{u,t}$ and X_u for notation simplicity. Since we build an ETD-SAC detector for each user with this user's historical electricity consumption data, this does not cause any ambiguity.

3.3.1. Series decomposition

As a common technique in time series analysis, time series decomposition can divide a time series into multiple components, each reflecting an underlying predictable pattern. This method is primarily helpful for analyzing historical changes over time. In classification tasks, series decomposition is often utilized as a preprocessing step for historical series before classification, as seen in approaches like Prophet [33], which uses trend-seasonality decomposition. However, the preprocessing method is constrained by its simplistic decomposition of historical series and does not capture the hierarchical interactions among underlying patterns in the long-term future. To address this problem, the ETD-SAC model incorporates the series decomposition layer as an internal operation module.

Let L denote the length of the input sequence. Let d denote the dimension of each element in the sequence. Let $X_{in} \in R^{L\times d}$ denote the input of the ETD-SAC detection. We take 96 electricity consumption measurements daily for seven consecutive days (i.e., one week) as the input. We have $L = 7$ and $d = 96$. The series decomposition layer separates electricity consumption data into seasonal and trend-cyclical components to account for the complex characteristics of electricity consumption patterns. Besides, we apply an average pooling process with the same filling strategy to perform the sliding average [17], extracting the trend-cyclical component. The seasonal component is then obtained by subtracting the trend-cyclical component from the electricity consumption data. The following formula describes this process.

$$\begin{aligned} X_t &= \text{AvgPool}(\text{Padding}(X_{in})) \\ X_s &= X_{in} - X_t, \end{aligned} \tag{3}$$

where $X_s \in R^{L\times d}$ is the seasonal part obtained by decomposing the time series, and $X_t \in R^{L\times d}$ is the trend-cyclical part obtained by decomposing the time series. Specifically, the $Padding(\cdot)$ adds extra zero elements to the edge of the input data to make the data obtained after subsequent pooling operations the same size as the input data. The $AvgPool(\cdot)$ takes the average of all elements in the pooling window and replaces the elements in the window with this average value. For the convenience of the subsequent introduction of the implementation details of the ETD-SAC detector, we represent the series decomposition module as $SeriesDecomp(\cdot)$.

3.3.2. Series embedding

The high-dimensional feature representation of electricity consumption data enables more effective identification of temporal dependencies within complex consumption patterns. Let d_{model} denote a user pre-specified target dimension to be converted. To transform the seasonal component $X_s \in R^{L\times d}$ into a specified high-dimensional space $R^{L\times d_{model}}$, the layer utilizes a convolutional layer for feature embedding. This process is expressed as

$$X_{conv} = W_{conv} \times X_s + B_{conv}, \tag{4}$$

where $X_{conv} \in R^{L\times d_{model}}$ represents the output of the convolutional layer, W_{conv} represents the convolution kernel, and B_{conv} is the bias term.

Integrating positional information with electricity consumption data enables more effective identification of temporal dependencies in electricity consumption patterns. The layer incorporates a positional encoding module described in Vaswani et al. [34] to achieve this. Let pos denote the position of an element within the input sequence X_{in}. The formula for generating the positional encoding is given as

$$\begin{aligned} PE_{pos,2i} &= \sin\left(\frac{pos}{10000^{2i/d_{model}}}\right) \\ PE_{pos,2i+1} &= \cos\left(\frac{pos}{10000^{2i/d_{model}}}\right), \end{aligned} \tag{5}$$

where $pos \in \{0, 1, \ldots, L-1\}$, $i \in \{0, 1, \ldots, \lfloor \frac{d_{model}-1}{2} \rfloor\}$ refers to the dimension index in the positional encoding. $\lfloor\cdot\rfloor$ represents the round down operation. Each dimension of the positional encoding is associated with a sinusoidal function, so we can obtain $PE_{pos} = \{PE_{pos,0}, PE_{pos,1}, \ldots, PE_{pos,d_{model}-1}\}$. This function was chosen based on the hypothesis that it would help the model effectively learn to focus on relative positions. This is because, for any fixed offset k, PE_{pos+k} can be represented as a linear function of PE_{pos}. Let PE denote the positional encoding vector. Then, the position encoding vector PE can be obtained as

$$PE = \begin{pmatrix} PE_0 \\ PE_1 \\ \vdots \\ PE_{L-1} \end{pmatrix}.$$

Let $X_{ems} \in R^{L\times d_{model}}$ denote the encoding series obtained by the series embedding layer. The following formula obtains it:

$$X_{ems} = X_{conv} + PE. \tag{6}$$

3.3.3. Feature exaction

The feature extraction layer primarily consists of the auto-correlation mechanism layer and the feedforward neural network layer.

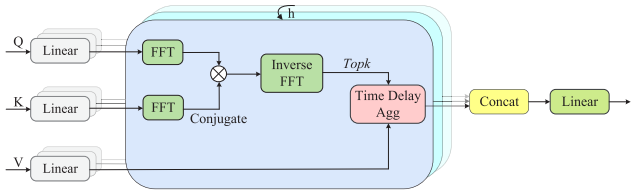


Fig. 4. The auto-correlation mechanism.

The auto-correlation mechanism layer: As illustrated in Fig. 4, the layer utilizes the multi-head auto-correlation mechanism to simultaneously focus on information from various projection spaces, extracting meaningful temporal dependencies from electricity consumption data.

Given time series Z_t, based on the Wiener–Khinchin theorem [35], autocorrelation $R_{ZZ}(\tau)$ can be calculated by Fast Fourier Transforms (FFT) as

$$\begin{aligned} S_{ZZ}(f) &= F(Z_t)F^*(Z_t) \\ &= \int_{-\infty}^{\infty} Z_t e^{-i2\pi tf}\, dt \overline{\int_{-\infty}^{\infty} Z_t e^{-i2\pi tf}\, dt} \\ R_{ZZ}(\tau) &= F^{-1}(S_{ZZ}(f)) = \int_{-\infty}^{\infty} S_{ZZ}(f) e^{i2\pi f\tau}\, df, \end{aligned} \tag{7}$$

where $\tau \in \{1, 2, \ldots, L\}$, F represents the FFT and F^{-1} represents its inverse. The operator $*$ denotes the conjugate operation, and $S_{ZZ}(f)$ is in the frequency domain. Note that the series autocorrelation of all lags in $\{1, 2, \ldots, L\}$ can be computed at once using FFT. Therefore, the autocorrelation has a computational complexity of $O(L \log(L))$.

The auto-correlation mechanism layer first employs h different linear projections to map the input data X_{ems} to three series query Q_i, key K_i and value V_i, $i \in \{1, 2, \ldots, h\}$, $Q_i, K_i, V_i \in R^{L \times \frac{d_{model}}{h}}$. For each set of (Q_i, K_i, V_i), the layer then calculates autocorrelation $R_{Q_iK_i}(\tau)$ between series Q_i and K_i, and selects the top k positions with the highest autocorrelation as the time delay starting point. Then, as shown in Eq. (8), we normalize the top k autocorrelation. At these top k starting points, a time delay operation is performed on V_i to generate a new time delay series. Subsequently, the autocorrelations and the time-delayed series are weighted and summed up to produce the output series H_i of a single auto-correlation mechanism. Finally, the layer concatenates the H_i and performs another linear projection to obtain the final result. Specifically, the auto-correlation mechanism is described as

$$\begin{aligned} \tau_1, \ldots, \tau_k &= \underset{\tau \in \{1, \ldots, L\}}{arg\,Topk} (R_{Q_iK_i}(\tau_k)) \\ \hat{R}_{Q_iK_i}(\tau_j) &= \frac{e^{R_{Q_iK_i}(\tau_j)}}{\sum_{j=1}^{k} e^{R_{Q_iK_i}(\tau_j)}} \\ H_i &= \sum_{j=1}^{k} \mathrm{Roll}(V_i, \tau_j) \hat{R}_{Q_iK_i}(\tau_j) \\ X_{auto} &= W_{muti} \times \mathrm{Concat}(H_1, \ldots, H_h), \end{aligned} \tag{8}$$

where $argTopk(\cdot)$ is to get the arguments of the top k autocorrelations and set $k = \lfloor c \cdot \log(L) \rfloor$, c is a hyper-parameter; $Roll(V, \tau)$ represents the operation to V with time delay τ, during which elements that are shifted beyond the first position are reintroduced at the last position. W_{muti} represents the parameter matrix. Concat$(\cdot)$ represents the concatenation operation of vectors. $X_{auto} \in R^{L \times d_{model}}$ is the output of the auto-correlation attention mechanism layer.

Then, we apply the series decomposition module to decompose further the complex consumption patterns, which can be expressed as follows:

$$X_{s,auto}, X_{t,auto} = SeriesDecomp(X_{auto} + X_{ems}), \tag{9}$$

where $X_{s,auto} \in R^{L \times d_{model}}$ is the seasonal part, and $X_{t,auto} \in R^{L \times d_{model}}$ is the trend-cyclical part.

To better capture the long-term temporal dependencies of the user's electricity consumption time series, we sequentially connect n auto-correlation mechanism layers in the feature extraction module.

The feedforward neural network layer: The feedforward neural network leverages nonlinear activation functions to learn intricate consumption patterns, allowing it to extract temporal dependencies from electricity consumption data effectively. Then this process can be expressed as follows:

$$X_{feed} = max(0, X_{s,auto} \times W_{1,feed} + b_{1,feed}) \times W_{2,feed} + b_{2,feed} \tag{10}$$

where $W_{1,feed}$, $W_{2,feed}$ are the parameter matrices. $b_{1,feed}$, $b_{2,feed}$ are the bias terms. $max(0, m)$ means taking the larger value of 0 and m. $X_{feed} \in R^{L \times d_{model}}$ is the output of the feedforward neural network layer.

Then, we also apply the series decomposition module to decompose the complex consumption patterns, which can be expressed as follows:

$$X_{s,feed}, X_{t,feed} = SeriesDecomp(X_{feed} + X_{s,auto}), \tag{11}$$

where $X_{s,feed} \in R^{L \times d_{model}}$ is the seasonal part, and $X_{t,feed} \in R^{L \times d_{model}}$ is the trend-cyclical part.

Finally, we superimposes the seasonal part with the cumulative trend-cyclical part as the output of the feature extraction module, which can be expressed as follows:

$$X_{out} = W_s \times X_{s,feed} + W_{t,feed} \times X_{t,feed} + W_{t,auto} \times X_{t,auto} + X_t \tag{12}$$

where W_s is the parameter matrix to project the deep transformed seasonal part to the original dimension. $W_{t,feed}$ and $W_{t,auto}$ are parameter matrices to project the deep transformed trend-cyclical part to the original dimension. $X_{out} \in R^{L \times d}$ is the output of the feature extraction module.

3.3.4. Classifier

The classifier consists of two consecutive convolutional layers designed to analyze temporal dependencies and detect abnormal consumption patterns. The classification outcomes are passed through an activation function to produce the output $Y_{out} \in R^{L \times 1}$ with values between 0 and 1, which can be expressed as follows:

$$Y_{out} = \sigma(W_{2,class} \times (W_{1,class} \times X_{out} + b_{1,class}) + b_{2,class}), \tag{13}$$

where $W_{1,class}$, $W_{2,class}$ are the parameter matrices. $b_{1,class}$, $b_{2,class}$ are the bias terms. $\sigma(z) = \frac{1}{1+e^{-z}}$ is the activation function.

Each value in Y_{out} is then compared to a predefined electricity consumption abnormality threshold, typically set at 0.5. If a value exceeds the threshold, the corresponding detection sample is classified as an electricity theft case; otherwise, it is considered benign.

4. Experiments

4.1. Experiment settings

In this paper, we conduct an extensive evaluation of the deployed ETD-SAC detector using the real-world consumption dataset [36], which includes the electricity consumption data of 370 residential and industrial users from 2011 to 2014, with a sampling interval of 15 min. The dataset used for model training and testing spans 1096 days, from 2012 to 2014. We exclude user data with more than 20% missing values, leaving the historical electricity consumption data of 321 users. Since all the data in the dataset comes from honest users [36], we simulate malicious user behavior by introducing attacks to manipulate the electricity consumption data.

We split the user's electricity consumption time series into equally sized benign detection samples. Each sample contains 96 data, corresponding to one day's electricity consumption. Half of the benign and malicious samples are tampered with to ensure the balance of benign and malicious samples. Let t_{beg} and t_{end} denote a detection sample's

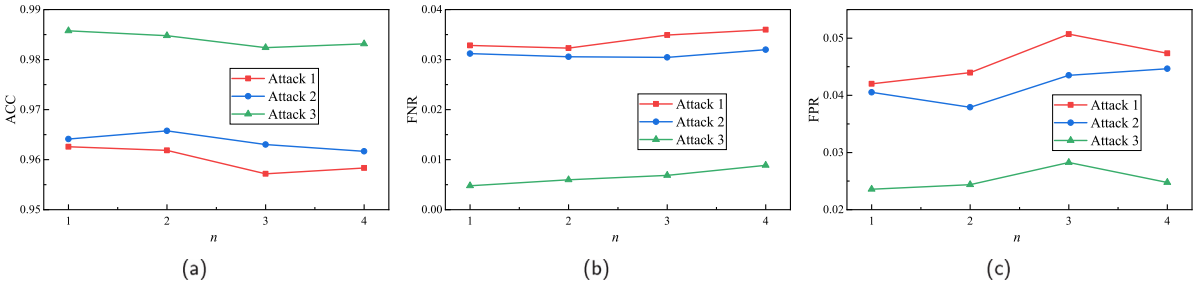


Fig. 5. Simulation results regarding how parameter *layer* affects the performance of the ETD-SAC detector, evaluated by: (a) ACC; (b) FNR; (c) FPR.

start and end measurement periods, respectively. By analyzing the behavior patterns of malicious users, we consider the following three types of attacks: (1) Attack 1: $x'_{u,t} = \alpha_t x_{u,t}$, $\forall t \in \{t_{beg}, t_{beg+1}, \ldots, t_{end}\}$, where $x'_{u,t}$ denotes the tth generated electricity consumption of malicious users. By Attack 1, malicious users tamper with electricity consumption by a constantly changed coefficient α_t. (2) Attack 2: $x'_{u,t} = \beta x_{u,t}$, $t \in \{t_{beg}, t_{beg+1}, \ldots, t_{end}\}$, by which malicious users tamper with electricity consumption by a constant coefficient β. (3) Attack 3: $x'_{u,t} = \begin{cases} x_{u,t}, \forall t \in \{t_{beg}, t_{beg}+1, \ldots, t_{ste}\}, \\ \alpha_t x_{u,t}, \forall t \in \{t_{ste}+1, t_{ste}+2, \ldots, t_{end}\} \end{cases}$, by which malicious users launch Attack 1 from some measurement periods indexed by t_{ste} between t_{beg} and t_{end}.

We split each user's detection samples into training, verification, and test sets at a ratio of 7:1:2. As mentioned before, we train an ETD-SAC detector for each user in the community and deploy them in the cloud server. These deployed ETD-SAC detectors are implemented in Pytorch. We set the batch size to 32 and used the binary cross-entropy loss for model training. The Adam optimizer is applied to update the model parameters with an initial learning rate of 10^{-4}, which decreases with the number of iterations. We add the dropout layer with a probability of 0.5 to prevent the model from overfitting. Additionally, we incorporate an early stopping mechanism, where training halts if the verification set loss does not decrease for three consecutive iterations. A total of 10 training iterations were conducted, with each experiment repeated three times. Let TP, TN, FP, and FN denote true positive, true negative, false positive, and false negative in the confusion matrix, respectively. We use the following three metrics to evaluate the performance of electricity theft detection: (1) Accuracy (ACC): the ratio of the number of correct predicted samples to the number of all samples, with $ACC = \frac{TP+TN}{TP+TN+FP+FN}$. (2) False negative rate (FNR): the ratio of the number of malicious samples mistakenly predicted as honest samples to the total number of malicious samples, with $FNR = \frac{FN}{TP+FN}$. (3) False positive rate (FPR): the ratio of honest samples mistakenly predicted as malicious to the total number of honest samples, with $FPR = \frac{FP}{FP+TN}$.

4.2. Parameter analysis

In this sub-section, we investigate the impact of parameters on the performance of the ETD-SAC detector. We conduct comparative experiments with various parameter values under Attacks 1, 2, and 3 to achieve optimal performance. The specific parameters for the attack models are as follows: We set α_t to range from 0.6 to 0.7 under Attack 1; We set $\beta = 0.6$ under Attack 2; Let γ denote the ratio of electricity theft periods to the total measurement periods; We set $\alpha_t \in (0.1, 0.9)$ and $\gamma = 0.4$ under Attack 3.

(1) *Experiment 1*: To investigate the impact of the number n of auto-correlation mechanism layers on the performance of the ETD-SAC detector, we conduct comparative experiments by setting the values of n to 1, 2, 3, and 4, respectively, with $c = 2$, $d_{model} = 1024$, and $h = 8$ under all attack models. The experimental results are shown in Fig. 5. When we set the value of n to 1 or 2 instead of 3 or 4, the detector performs better with higher ACC, lower FNR, and FPR under all attack models. Compared to when the value of n is set to 1, setting the value of n to 2 degrades the detection performance under Attacks 1 and 3 but significantly improves performance under Attack 2. Setting the value of n to 2 yields optimal comprehensive detection performance. As the value of n increases, the receptive field of the detector enlarges, allowing it to better capture long-range dependencies in electricity consumption data. However, the experimental results show that when the value of n is large, the model exhibits increased complexity, potentially resulting in redundant or noisy feature extraction and overfitting. Conversely, if the value of n is too small, the effective receptive field becomes inadequate for capturing critical long-term dependencies. Therefore, in this study, we set $n = 2$.

(2) *Experiment 2*: To investigate the impact of the hyperparameter c for obtaining top k on the performance of the ETD-SAC detector, where $k = [c \times \log L]$ indicates the number of delay sequences $X_{t-\tau}$, we conduct comparative experiments by setting the values of c to 1, 2, 3, and 4, respectively, with $n = 2$, $d_{model} = 1024$, and $h = 8$ under all attack models. The experimental results are shown in Fig. 6. With changes in the value of c, the detector's ACC, FNR, and FPR show no significant changes under all attack models, and the performance remains stable. Although increasing the value of c increases the number of delay sequences k, the additional information gained tends to saturate due to the small L of the input data. If the essential long-term dependencies are already captured with approximately $\lfloor c \times log L \rfloor$ sequences, then setting a larger value of c will not lead to substantial performance improvements but will result in higher computational cost and complexity. Therefore, in this study, we set $c = 2$.

(3) *Experiment 3*: To investigate the impact of the output dimension d_{model} of the embedding layer used for seasonal features in the series embedding module on the performance of the ETD-SAC detector, we conduct comparative experiments by setting the values of d_{model} to 128, 256, 512, and 1024, respectively, with $n = 2$, $c = 2$, and $h = 8$ under all attack models. The experimental results are shown in Fig. 7. With the increase in the value of d_{model}, the ACC of the detector shows an upward trend, while FNR and FPR show a downward trend under all attack models, and the performance gradually improves. When the value of d_{model} is set to 1024, the detection performance slightly decreases under Attack 3. A higher embedding dimension allows the model to capture more nuanced and fine-grained seasonal patterns, enhancing its ability to learn the complex dependencies in electricity consumption data. However, according to the principle of diminishing returns, increasing the value of d_{model} beyond a certain point introduces redundancy and even leads to overfitting, as the model starts to learn noise rather than meaningful patterns. Moreover, higher dimensions inherently increase computational cost and model complexity. Therefore, in this study, we set $d_{model} = 1024$.

(4) *Experiment 4*: To investigate the impact of the dimension *head* of the linear projection in the multi-head auto-correlation mechanism on the performance of the ETD-SAC detector, we conduct comparative experiments by setting the values of h to 4, 8, 16, and 32, respectively, with $n = 2$, $c = 2$, and $d_{model} = 1024$ under all attack models. The experimental results are shown in Fig. 8. With changes in the value of h, the detector's ACC, FNR, and FPR show no significant changes under

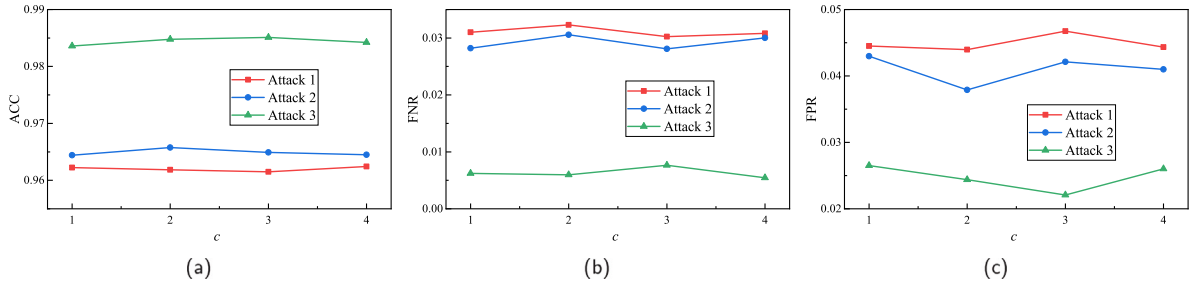


Fig. 6. Simulation results regarding how parameter c affects the performance of the ETD-SAC detector, evaluated by: (a) ACC; (b) FNR; (c) FPR.

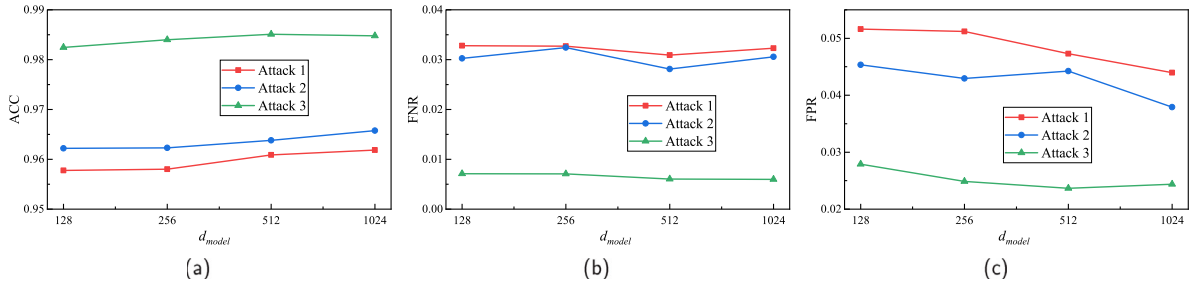


Fig. 7. Simulation results regarding how parameter d_{model} affects the performance of the ETD-SAC detector, evaluated by: (a) ACC; (b) FNR; (c) FPR.

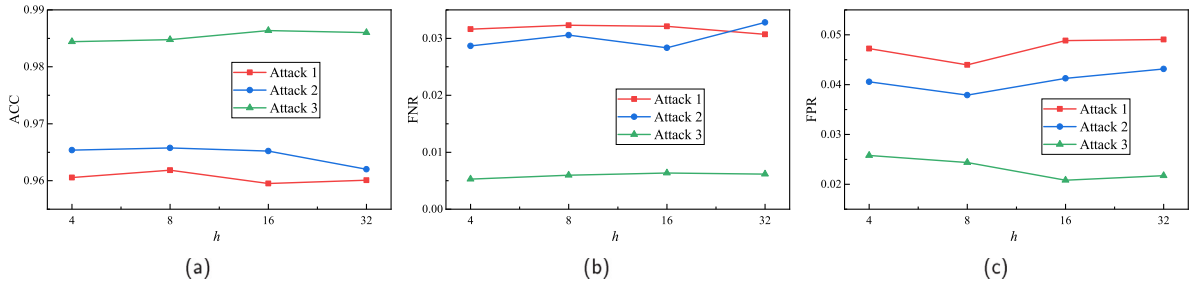


Fig. 8. Simulation results regarding how parameter h affects the performance of the ETD-SAC detector, evaluated by: (a) ACC; (b) FNR; (c) FPR.

Attack 3, and the performance remains stable. When we set the value of h to 4 or 8 instead of 16 or 32, the detector performs better with higher ACC, lower FNR, and FPR under Attack 1 and 2. Compared to when the value of h is set to 4, setting the value of h to 8 improves the detection performance under Attack 1 and 2. Setting the value of h to 8 yields optimal comprehensive detection performance. The value of h determines the number of parallel projection heads, indicating that the total embedding dimension d_{model} is divided among h heads. With $d_{model} = 1024$, setting $h = 8$ results in an allocation of 128 dimensions per head. This allocation is widely regarded as optimal for balancing the expressiveness and diversity of feature representations. A smaller value of h increases the dimensionality of each head, which may result in less specialized subspaces and a diminished ability to capture distinct features. Conversely, a larger value of h reduces the per-head dimensionality to 64 or 32, respectively, potentially constraining the ability of each head to learn meaningful patterns and substantially increasing the total parameter count, thereby heightening the risk of overfitting. Therefore, in this study, we set $h = 8$.

(5) *Experiment 5*: We replace the auto-correlation mechanism module in the ETD-SAC detector with the self-attention mechanism and conduct comparative experiments under all attack models. Following the parameter analysis, we set the parameter values of the ETD-SAC detector as follows: $n = 2$, $c = 2$, $d_{model} = 1024$, and $h = 8$. The experimental results are shown in Fig. 9. Compared with the detector based on the self-attention mechanism, using the auto-correlation mechanism significantly improved ACC. It reduced FNR and FPR, verifying the effectiveness of series-wise connections compared to point-wise self-attention. The FNR of the detector based on the self-attention mechanism is substantially higher than the FPR. It tends to misclassify malicious users as benign users, whereas the detector based on the auto-correlation mechanism distinguishes malicious users from benign users more effectively.

4.3. The ETD-SAC detector v.s. other detectors

In this section, we verify the effectiveness of the ETD-SAC detector by conducting various experiments under all attack models and comparing its performance with the following five baseline detectors: (1) the WDCNN detector [27], (2) the CNN-LSTM detector [28], (3) the LFPR-DNN detector [25], and (4) the HORLN detector [26]. We set the parameter values of the baseline detectors according to the values reported in the original studies. Following the parameter analysis, we set the parameter values of the ETD-SAC detector as follows: $n = 2$, $c = 2$, $d_{model} = 1024$, and $h = 8$.

(1) *Experiment 1*: We assume that malicious users launch Attack 1. Specifically, we consider the following scenarios to compare electricity theft detection performance: (1) Case I: $\alpha_t \in (0.6, 0.7)$; (2) Case II: $\alpha_t \in (0.7, 0.8)$; (3) Case III: $\alpha_t \in (0.8, 0.9)$. The experimental results are shown in Fig. 10. As α_t increases, the ACC of all detectors exhibits a decreasing trend, while the FNR and FPR exhibit an increasing trend. As common knowledge, the smaller the electricity malicious users steal, the more difficult it is to detect their theft behavior. Compared to baseline detectors, the ETD-SAC detector achieves the highest ACC and the lowest FNR in all scenarios. In Cases I and II, compared to the HORLN detector,

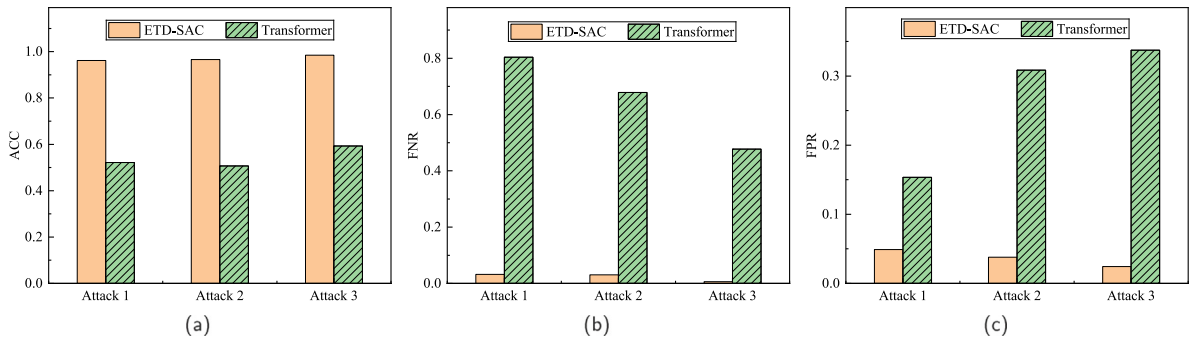


Fig. 9. Comparing the auto-correlation mechanism with the self-attention mechanism in the ETD-SAC detector, evaluated by: (a) ACC; (b) FNR; (c) FPR.

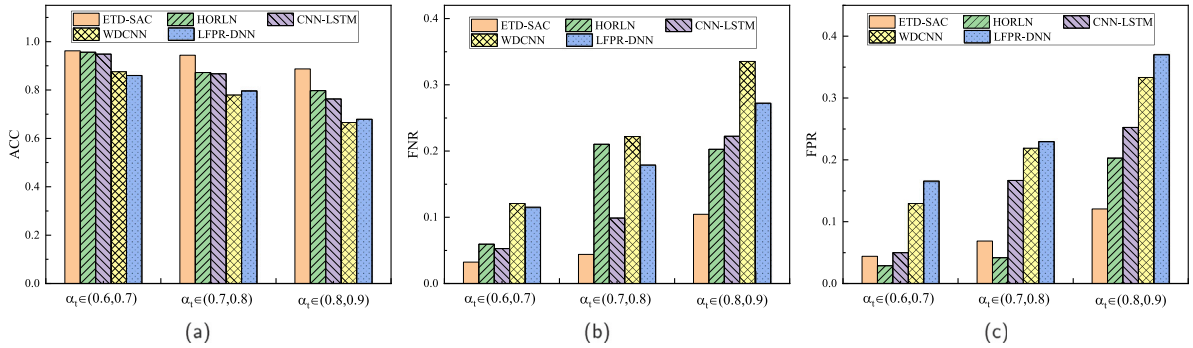


Fig. 10. Comparing the proposed ETD-SAC detector with state-of-the-art detectors under Attack 1, evaluated by: (a) ACC; (b) FNR; (c) FPR.

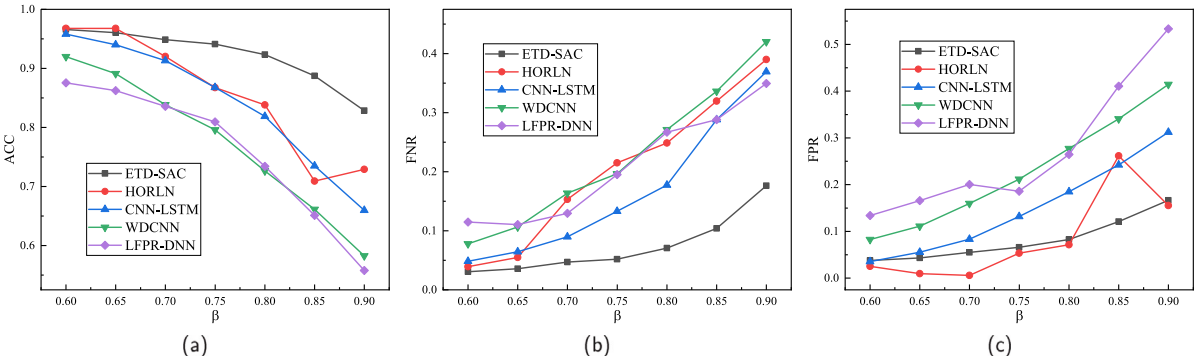


Fig. 11. Comparing the proposed ETD-SAC detector with state-of-the-art detectors under Attack 2, evaluated by: (a) ACC; (b) FNR; (c) FPR.

the ETD-SAC detector exhibits a significantly lower FNR and a slightly higher FPR. The HORLN detector tends to misclassify malicious users as benign users, whereas the ETD-SAC detector distinguishes malicious users from benign users more effectively.

(2) *Experiment 2*: We assume that malicious users launch Attack 2. We set the value of β to range from 0.6 to 0.9 with an interval of 0.05 to compare electricity theft detection performance. The experimental results are shown in Fig. 11. As β increases, the amount of electricity stolen by malicious users decreases. Consequently, the ACC of all detectors shows a decreasing trend, while the FNR and FPR show an increasing trend. Compared to the baseline detectors, except for the HORLN detector, the ETD-SAC achieves the highest ACC and the lowest FNR and FPR in all scenarios. When β is less than 0.7, the ACC of the ETD-SAC detector and the HORLN detector are not much different, and their detection performance is comparable. When β is greater than or equal to 0.7, the ACC of the ETD-SAC detector is significantly higher than that of the HORLN detector. When malicious users steal a small amount of electricity, the ETD-SAC detector demonstrates excellent detection performance.

(3) *Experiment 3*: We assume that malicious users launch Attack 3. We set the value of γ to range from 0.1 to 0.4, with an interval of 0.05, to compare electricity theft detection performance. The experimental results are shown in Fig. 12. As γ increases, the ACC of all detectors shows an increasing trend, while the FNR and FPR show a decreasing trend. As common knowledge, the shorter the time for malicious users to steal electricity, the more difficult it is to detect their theft behavior. When γ is less than 0.2, malicious users steal electricity for a short period, significantly reducing the detection performance, especially the WDCNN detector, which struggles to identify short-term electricity theft behavior from long-term electricity consumption data. The CNN-LSTM detector achieves the highest ACC and the lowest FNR and FPR, which first employs the CNN network for local perception, effectively capturing local anomaly features in electricity consumption data. When γ is greater than or equal to 0.2, the ACC, FNR, and FPR of the ETD-SAC and CNN-LSTM detectors differ slightly, and their detection performance is comparable. When malicious users steal electricity for an extended period, the ETD-SAC detector demonstrates excellent detection performance.

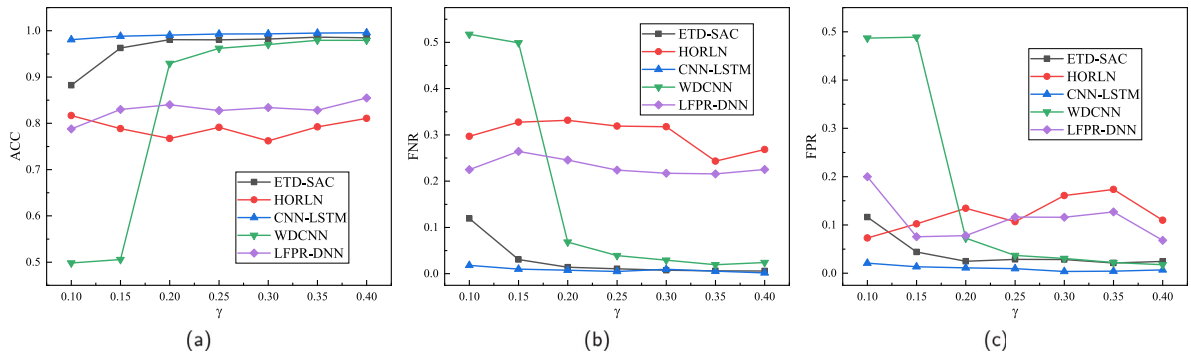


Fig. 12. Comparing the proposed ETD-SAC detector with state-of-the-art detectors under Attack 3, evaluated by: (a) ACC; (b) FNR; (c) FPR.

5. Conclusion

In this paper, we propose a new deep learning-based electricity theft detection framework in the context of cloud computing to identify malicious users in smart grids. The deployed ETD-SAC detector on the cloud server progressively decomposes consumption patterns to highlight the inherent properties of electricity consumption time series and aggregates the dependencies representation at the subsequence level based on the series periodicity. In this way, the detector effectively extracts reliable long-range dependencies to identify abnormal consumption patterns and detect electricity theft behaviors. We have conducted extensive experiments to evaluate the performance of the ETD-SAC detector, and the results show that the approach has excellent performance in terms of accuracy, false negative rate, and false positive rate. In addition, the cloud server provides a substantial storage capacity and robust computing power, which allows us to economically and efficiently identify electricity theft users.

CRediT authorship contribution statement

Zhen Si: Writing – original draft, Methodology. **Zhaoqing Liu:** Writing – original draft, Methodology. **Changchun Mu:** Software, Formal analysis, Data curation. **Meng Wang:** Project administration, Investigation. **Tongxin Gong:** Formal analysis, Data curation. **Xiaofang Xia:** Writing – review & editing, Validation, Investigation. **Qing Hu:** Writing – review & editing, Investigation. **Yang Xiao:** Writing – review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China under Grant 62372360; in part by the Youth Elite Scientist Sponsorship Program by the China Association for Science and Technology under Grant YESS20220610; in part by the Innovation Capability Support Program of Shaanxi Province under Grant 2024ZC-KJXX-021; in part by the Shenzhen Science and Technology Program under Grant KCXFZ20211020174801002; in part by the Natural Science Basic Research Program of Shaanxi under Grant 2023-JC-YB-558; in part by the Scientific Research Program Funded by Shaanxi Provincial Education Department under Grant 23JS028; and partly by the Scientific and Technological Program of Xi'an under Grant 24GXFW0016.

Data availability

Data will be made available on request.

References

[1] R. Deng, C.-W. Ten, C. Li, D. Niyato, F. Teng, Guest editorial: Introduction to special issue on "cloud-edge-end orchestrated computing for smart grid", IEEE Trans. Cloud Comput. 11 (2) (2023) 1107–1110.
[2] S. Tuli, F. Mirhakimi, S. Pallewatta, S. Zawad, G. Casale, B. Javadi, F. Yan, R. Buyya, N.R. Jennings, AI augmented edge and fog computing: Trends and challenges, J. Netw. Comput. Appl. 216 (2023) 103648.
[3] Q. Duan, J. Huang, S. Hu, R. Deng, Z. Lu, S. Yu, Combining federated learning and edge computing toward ubiquitous intelligence in 6G network: Challenges, recent advances, and future directions, IEEE Commun. Surv. Tutor. 25 (4) (2023) 2892–2950.
[4] M. Hatami, M.A. Nasab, Y. Chen, J. Mohammadi, E. Ardiles-Cruz, E. Blasch, ELOCESS: An ESS management framework for improved smart grid stability and flexibility, IEEE Trans. Consum. Electron. (2024).
[5] X. Xu, R. Mo, F. Dai, W. Lin, S. Wan, W. Dou, Dynamic resource provisioning with fault tolerance for data-intensive meteorological workflows in cloud, IEEE Trans. Ind. Inform. 16 (9) (2019) 6172–6181.
[6] Financial Bin, How cloud computing saves time and money, 2021, https://financialbin.com/how-cloud-computing-saves-time-and-money/.
[7] Standleys, How the cloud saves businesses money, 2024, https://www.standleys.com/blog/how-the-cloud-saves-businesses-money.
[8] NES, Energy theft and fraud reduction, 2020, URL https://www.smart-energy.com/industry-sectors/energy-grid-management/energy-theft-and-fraud-reduction/.
[9] T. Sharma, K. Pandey, D. Punia, J. Rao, Of pilferers and poachers: Combating electricity theft in India, Energy Res. Soc. Sci. 11 (2016) 40–52.
[10] C. Gladwin, How to deal with electricity theft, 2013, URL https://www.property24.com/articles/how-to-deal-with-electricity-theft/18742.
[11] X. Xia, Y. Xiao, W. Liang, J. Cui, Detection methods in smart meters for electricity thefts: A survey, Proc. IEEE 110 (2) (2022) 273–319.
[12] Z. Yan, H. Wen, Performance analysis of electricity theft detection for the smart grid: An overview, IEEE Trans. Instrum. Meas. 71 (2021) 1–28.
[13] X. Xia, J. Lin, Q. Jia, X. Wang, C. Ma, J. Cui, W. Liang, ETD-ConvLSTM: A deep learning approach for electricity theft detection in smart grids, IEEE Trans. Inf. Forensics Secur. 18 (2023) 2553–2568.
[14] W. Du, D. Côté, Y. Liu, Saits: Self-attention-based imputation for time series, Expert Syst. Appl. 219 (2023) 119619.
[15] S. Chen, Y. Yang, S. You, W. Chen, Z. Li, A study of electricity theft detection method based on anomaly transformer, in: The 11th CCF Conference on BigData, Nanjing, China, 2023, pp. 164–180.
[16] J. Shi, Y. Gao, D. Gu, Y. Li, K. Chen, A novel approach to detect electricity theft based on conv-attentional transformer neural network, Int. J. Electr. Power Energy Syst. 145 (2023) 108642.
[17] H. Wu, J. Xu, J. Wang, M. Long, Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting, Adv. Neural Inf. Process. Syst. 34 (2021) 22419–22430.
[18] Z. Si, Z. Liu, C. Mu, X. Xia, ETD-SAC: A series-wise auto-correlation mechanism based electricity theft detector for smart grids, in: The 2nd International Conference on Data Security and Privacy Protection, DSPP 2024, Xi'an, China, 2024, pp. 255–266.
[19] Y. Zhou, Y. Liu, S. Hu, Energy theft detection in multi-tenant data centers with digital protective relay deployment, IEEE Trans. Sustain. Comput. 3 (1) (2017) 16–29.

[20] X. Xia, Y. Xiao, W. Liang, M. Zheng, GTHI: A heuristic algorithm to detect malicious users in smart grids, IEEE Trans. Netw. Sci. Eng. 7 (2) (2018) 805–816.
[21] W. Han, Y. Xiao, NFD: Non-technical loss fraud detection in smart grid, Comput. Secur. 65 (2017) 187–201.
[22] X. Xia, J. Lin, Y. Xiao, J. Cui, Y. Peng, Y. Ma, A control-chart-based detector for small-amount electricity theft (SET) attack in smart grids, IEEE Internet Things J. 9 (9) (2021) 6745–6762.
[23] M. Ismail, M.F. Shaaban, M. Naidu, E. Serpedin, Deep learning detection of electricity theft cyber-attacks in renewable distributed generation, IEEE Trans. Smart Grid 11 (4) (2020) 3428–3437.
[24] X. Cui, S. Liu, Z. Lin, J. Ma, F. Wen, Y. Ding, L. Yang, W. Guo, X. Feng, Two-step electricity theft detection strategy considering economic return based on convolutional autoencoder and improved regression algorithm, IEEE Trans. Power Syst. 37 (3) (2021) 2346–2359.
[25] D. Gu, Y. Gao, K. Chen, J. Shi, Y. Li, Y. Cao, Electricity theft detection in AMI with low false positive rate based on deep learning and evolutionary algorithm, IEEE Trans. Power Syst. 37 (6) (2022) 4568–4578.
[26] Y. Zhu, Y. Zhang, L. Liu, Y. Liu, G. Li, M. Mao, L. Lin, Hybrid-order representation learning for electricity theft detection, IEEE Trans. Ind. Inform. 19 (2) (2023) 1248–1259.
[27] Z. Zheng, Y. Yang, X. Niu, H.-N. Dai, Y. Zhou, Wide and deep convolutional neural networks for electricity-theft detection to secure smart grids, IEEE Trans. Ind. Inform. 14 (4) (2017) 1606–1615.
[28] M.N. Hasan, R.N. Toma, A.A. Nahid, M.M. Islam, J.M. Kim, Electricity theft detection in smart grid systems: A CNN-LSTM based approach, Energies 12 (17) (2019) 3310.
[29] A. Asghari, M.K. Sohrabi, Server placement in mobile cloud computing: A comprehensive survey for edge computing, fog computing and cloudlet, Comput. Sci. Rev. 51 (2024) 100616.
[30] C. Genes, I. Esnaola, S.M. Perlaza, L.F. Ochoa, D. Coca, Recovering missing data via matrix completion in electricity distribution systems, in: 2016 IEEE 17th International Workshop on Signal Processing Advances in Wireless Communications, SPAWC 2016, IEEE, Edinburgh, United Kingdom, 2016, pp. 1–6.
[31] J. Li, M. Yang, F.L. Lewis, M. Zheng, Compensator-based self-learning: Optimal operational control for two-time-scale systems with input constraints, IEEE Trans. Ind. Inform. (2024).
[32] Z. Xiao, Y. Xiao, D.H.-C. Du, Exploring malicious meter inspection in neighborhood area smart grids, IEEE Trans. Smart Grid 4 (1) (2013) 214–226.
[33] S.J. Taylor, B. Letham, Forecasting at scale, Amer. Statist. 72 (1) (2018) 37–45.
[34] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, Ł. Kaiser, I. Polosukhin, Attention is all you need, Adv. Neural Inf. Process. Syst. 30 (2017).
[35] N. Wiener, Generalized harmonic analysis, Acta Math. 55 (1) (1930) 117–258.
[36] A. Trindade, Electricity load diagrams 2011–2014, 2015, URL https://archive.ics.uci.edu/dataset/321/electricityloaddiagrams20112014.